UNIVAC SPLS

SYSTEMS PROGRAMMING LIBRARY SERVICES

- *Automatic Programming Systems*
- *Generalized Programming Systems*
- *Copies of Program Tapes and Decks*
- *Programming Information Exchange*
- *Technical Program Documentation*

September 21, 1965

UNIVAC 1050 MASS STORAGE SYSTEM
Programming Information Exchange
Bulletin 5

UNIVAC 1050 Mass Storage System P. I. E. Bulletin 5 announces the release and availability of UNIVAC 1050 Mass Storage System Preliminary Fastrand Software Description, UP 3951. 5, 17 pages.

This document provides a preliminary description of software routines that will enable the user to establish, maintain, and access files on FASTRAND drum(s). Flexibility is provided to allow the user to take advantage of certain hardware capabilities (e.g. Fastbands and the lockout feature). Software utility routines providing for loading and maintaining object programs are also described.

Copies of UNIVAC 1050 Mass Storage System Preliminary Fastrand Software Description, UP 3951.5, are being attached to this P.I.E. Bulletin and stocked in Holyoke, Massachusetts. Additional copies may be ordered through your local UNIVAC Manager.

MANAGER,
Systems Programming Library Services

TO LISTS: 10 U, 211, 630, 650, 692, and 153, page i only.

ATTACHMENT: UNIVAC 1050 Mass Storage System Preliminary Fastrand Software Description, UP 3951.5, plus page i, to S.P.L.S. Lists 34, 35 and 36.

UNIVAC DIVISION OF SPERRY RAND CORPORATION, SPERRY RAND BUILDING, NEW YORK, N. Y. 10019

PRELIMINARY FASTRAND SOFTWARE DESCRIPTION

UP 3951.5

1.0 C O N T E N T S

		Page
1.0	CONTENTS	
2.0	GENERAL DESCRIPTION	1
3.0	HARDWARE CONFIGURATION	2
	3.1 DRUM(S)	2
	3.2 MAGNETIC TAPE UNIT(S)	2
	3.3 CARD READER	2
	3.4 PRINTER	3
	3.5 PUNCH	3
	3.6 COMMUNICATIONS	3
	3.7 READ-PUNCH	3
	3.8 PAPER TAPE	3
4.0	DRUM FORMAT	4
	4.1 GENERAL	4
	4.2 RESERVED DRUM AREA	5
	4.3 SYSTEMS DIRECTORY	5
	4.4 FILE DIRECTORY	6
	4.5 PROGRAM DIRECTORY	7
5.0	DATA FORMAT	9
6.0	SOURCE CODE FORMAT	10
	6.1 SOURCE PROGRAM OR PROC HEADER SECTOR	10
	6.2 SOURCE PROGRAM OR PROC SENTINEL SECTOR	10
	6.3 SOURCE CODE SECTOR	10
7.0	OBJECT CODE FORMAT	11
	7.1 PROGRAM HEADER SECTOR	11
	7.2 R SECTOR	11
	7.3 S SECTOR	11
	7.4 INSTRUCTION SECTOR	12
	7.5 T SECTOR	12
	7.6 LDKEY SECTOR	12
	7.7 PROGRAM SENTINEL SECTOR	13
8.0	SOFTWARE ROUTINES	14
	8.1 INITIAL DRUM SETUP (IDMS)	14
	8.2 DRUM LOCATOR LOADER (DLL)	14
	8.3 SYSTEMS AREA DRUM LOAD (SADL)	14
	8.4 DRUM EXECUTIVE ROUTINE (OPRF)	14
	8.5 PAL DRUM ASSEMBLER	14
	8.6 AJAXF AND OPUSF	15
	8.7 FASTRAND DUMP (FRDP)	15
	8.8 FASTRAND LOAD (FRLD)	15
	8.9 DRUM SORT	15
	8.10 FILE CONTROL	15

This document is preliminary in nature and is intended as a vehicle for meeting immediate needs with regard to system familiarization and orientation. UNIVAC® Division of Sperry Rand Corporation reserves the right to change and/or modify such information contained herein as may be required by subsequent system developments.

2.0 GENERAL DESCRIPTION

The purpose of the FASTRAND software discussed in the following pages is to provide facilities for the FASTRAND user that will enable him to establish, maintain, and access files on FASTRAND drum(s). Software utility routines will be provided for loading and maintaining object and source programs on the drum(s). A FASTRAND Sort generator and assembler will also be provided.

Flexibility will be provided that will allow the user to allocate his files on a FASTRAND drum or drums, and take advantage of certain hardware capabilities (e.g. Fastbands and the lockout write feature). The software will handle any number of drums, and any allowable combination of the different types of drum (Fastrand I, Fastrand II or Modular FASTRAND) capable of being connected to a single UNIVAC 1050 Computer. The user must have an understanding of the FASTRAND hardware, and the FASTRAND hardware addressing scheme, before establishing conventions concerning the layout of the FASTRAND drum for his particular installation.

The FASTRAND software will operate with drums that have been laid out in accordance with a system of directories. These directory entries will contain the information required by the software routines to locate all files on the drum. FASTRAND users must, therefore, conform to the directory conventions in order to use the FASTRAND software.

The user will have the ability to place all software routines in any contiguous area of a drum that he desires with one exception. The first few sectors on drum unit zero are reserved for the software.

Conventions will be established regarding the physical appearance of date files, object programs, and source programs on the FASTRAND drum(s).

3.0 HARDWARE CONFIGURATION

3.1 DRUM(S)

At least one of the different types of drum available is required. The software will operate with any of the following drum combinations (maximum of 8 drum units).

3.1.1 FASTRAND I

64 sectors per head position
64 heads
96 tracks per head

3.1.2 FASTRAND II

64 sectors per head position
64 heads
192 tracks per head

3.1.3 Modular FASTRAND

32 sectors per head position
16 heads
128 tracks per head
1 master unit (minimum)
1 to 7 slaves (subunits per master)

3.1.4 Fastbands

The Fastband option can be added to any of the preceding drum types.

3.2 MAGNETIC TAPE UNIT(S)

Optional.

3.2.1 IIIA

3.2.2 IIIC

3.2.3 VIC

3.3 CARD READER

Required.

3.3.1 80 column Column Reader

3.3.2 80 column Row Reader

3.3.3 90 column Column Reader

3.3.4 90 column Row Reader

3.3.5 1004 - 80 or 90 column online to the UNIVAC 1050 (Channel 7)

3.4 PRINTER

Required.

3.4.1 Buffered

3.4.2 Unbuffered

3.4.3 1004 online to 1050 (standard 1050 code wheel)

3.5 PUNCH

Optional.

3.5.1 80 column

3.5.2 90 column

3.5.3 1004 - 80 or 90 column online to the UNIVAC 1050

3.6 COMMUNICATIONS*

Optional.

3.7 READ-PUNCH*

Optional.

3.8 PAPER TAPE*

Optional.

---

* Will be considered for programs operating with the OPR Executive Routine.

4.0 DRUM FORMAT

4.1 GENERAL

All files on the drum(s) will be located via a system of directories. These directories will define all existing files that are available to the software system as well as the physical drum characteristics pertaining to the drum addresses of these files. Since the physical drum characteristics will vary widely from installation to installation, it will be necessary for the user to inform the software of the kind and number of drums on his particular system. A maximum of eight drum units can be placed on a UNIVAC 1050. The kinds of drums available to the system are FASTRAND I, FASTRAND II and Modular FASTRAND which has one master and up to seven slaves. Options that are available on some or all of the different kinds of drums are: 1) Fastbands, 2) a hardware switch that prevents writing on certain portions (referred to as lock-out area) of a FASTRAND drum, and 3) the normal drum (that which is not included in the two above mentioned categories).

A facility will be provided that will enable the user to define the kinds and number of drums on his system and further subdivide his system to take advantage of any optional FASTRAND equipment that he might have. The definition of a user drum configuration and further subdivision of drum units must be established by using a utility program (see Initial Drum Setup Routine, Section 8.1) before any other software routines can reference information on a drum(s). The main function of the Initial Drum Setup Routine (IDMS) will be to establish a first level directory, known as the Systems Directory. The user will thus be able to request drum allocation within any of the classes of drum area as defined by the Systems Directory. The user must define a class of drum area which is reserved for the Systems Programming File, a Processor Scratch Area and User Program File(s). These areas must be defined with names the first two characters of which are unique to the system.

The source and object code utility routines (AJAXF and OPUSF) will maintain the information in these areas. A Systems Directory entry will be written by the IDMS routine based upon the parameter cards submitted by the user. The same name written in the Systems Directory entry will also be written in the File Directory entry. Names prefixed with an asterisk (*) will be treated in a unique manner by the software. The drum area defined by names prefixed with an asterisk (*) will be known as systems areas (see Section 4.5).

Each class of drum area is divided into two areas: 1) a File Directory and 2) a File area. The purpose of the File Directory is to identify specific files within the various classes of drum area. The size of each File Directory is defined at the same time the classes are defined. Any new File which is opened must be qualified by a particular Systems Directory name.

A class that is being maintained by AJAXF or OPUSF will contain in its file area two File Directory entries. One entry will describe the size of the File Directory and one entry will describe the File associated with this class. The remainder of the File area is divided into two areas: 1) a Program Directory and 2) a Program Area. The purpose of the Program Directory is to identify source and/or object programs that exist within a particular file. The Program Directory size will be defined by AJAXF by means of a parameter card.

A file that is not being maintained by the drum utility routines will contain data in its File area. These Files will not have a subordinate directory.

4.2 RESERVED DRUM AREA

The first few sectors on drum unit zero, starting at track zero, head zero, sector zero, will be reserved for a program that will locate and load absolute programs from the drum. The primary purpose of this program will be to locate and load an Executive Routine. This program will be written on the drum by the Initial Drum Setup Routine.

4.3 SYSTEMS DIRECTORY

The Systems Directory entries will immediately follow the locator-loader program on the drum, and will be written in consecutive drum addresses for a variable number of sectors, depending on the number of SD entries required. The first SD entry will describe the Systems Directory and will be known as the Systems Directory Descriptor (SDD). The Initial Drum Setup Routine will accept parameter cards and write the Systems Directory on the drum.

The format for the SDD is as follows:

Characters	Content
0 - 7	SYSDIRCT
8 - 11	Drum address of first SD entry.
12 - 15	Drum address of first sector beyond the area reserved for SD entries.
16 - 19	Drum address of the first free sector within the SD available for additional SD entries.

The format for the SD entries is as follows:

Characters	Content
0 - 7	Name
8 - 11	Drum address of the first sector of this class of drum area.
12 - 15	Drum address of the first sector beyond this class of drum area.
16	Modular FASTRAND Indicator
	0 = other than Modular FASTRAND
	1 = Modular FASTRAND

Each Systems Directory entry must describe a contiguous area of drum storage. Systems Directory entries can not overlap, i.e., a drum address associated with one entry may not be associated with another

entry. The Systems Directory entries need not describe the entire drum storage available, i.e., an area of drum storage may be reserved that is not included for use by the software.

4.4 FILE DIRECTORY

The first n sectors of each file area will be reserved for a File Directory. The number of sectors reserved for this purpose (n) will be specified by the user and will depend on the maximum number of files to be contained in a particular class (indicated by an SD entry) of area. Each SD entry will point directly to a File Directory. The first FD entry will describe its File Directory and will be known as a File Directory Descriptor (FDD). FD entries describing specific files will immediately follow an FDD entry. The FDD entry is written by the IDMS routine and its contents are established based on the user specification. The IDMS routine will write a single FD entry for classes of areas which are reserved for source and/or object programs (indicated by names that are uniquely prefixed).

FD entries for data files will be written by opening new files using the file control routines.

The format of the FDD entry is as follows:

Characters	Content
0 - 7	FLEDIRCT
8 - 11	Drum address of first FD entry.
12 - 15	Drum address of first sector beyond the area reserved for FD entries.
16 - 19	Drum address of first free sector within the FD available for additional FD entries.

The format of the FD entry is as follows:

Characters	Content
0 - 7	Name
8 - 11	Drum address of the first item in the file.
12 - 15	Drum address of the first sector beyond the file area.
16 - 19	Drum address of the next free sector available within the file area.
20 - 23	Drum address of the next free sector available within the file area for secondary chain elements.
24	Head position indicator.
	1 = File must start at the beginning of a head position.
	0 = File does not have to start at the beginning of a head position.

Each FD entry must describe a contiguous area of drum storage. The boundaries of a file must be contained within the drum area described by a single SD entry. The drum storage allocated to one file will remain allocated to this file until the file is closed and its area released for another file.

4.5 PROGRAM DIRECTORY

Entries in the SD and FD with uniquely prefixed names designating systems areas will reserve contiguous drum area for the Systems Programming File, User Program Files, and a Scratch Area. In order to maintain these programs, it is necessary to establish a third level of directory known as the Program Directory (PD). The Program Directories will be established and maintained by AJAXF and OPUSF. Identification of a systems area that is to be referenced by AJAXF, OPUSF, the Executive Routine, and the PAL Drum Assembler will be indicated by the first two characters of a name that was assigned to the systems area.

Systems areas that contain source and/or object code programs or a Scratch area must have names in the following format:

Characters		Content
0	*	Indicates systems area.
1	*	Indicates systems programming file.
	$	Indicates scratch area.
	A - Z	Any alphabetic character indicates a user program file.
2 - 7		Will contain any characters and comprise the remainder of the name.

An asterisk (*) in position Ø of a class name indicates a systems area definition. Position 1 of a name that is defined with an asterisk in position Ø is used as a systems area identifier, i.e., it identifies a specific systems area. The systems area identifier will be used by the Executive Routine and the Drum Locator/Load (DLL) to locate object programs on a drum.

The first sector of the PD will describe the PD, i.e., the boundaries of a PD. This sector will be known as the Program Directory Descriptor (PDD). The number of sectors reserved for PD entries will be defined by the user through AJAXF. The next sector and all other sectors in the PD will describe the routines contained in this file. All addresses in the PD and PDD will be file relative.

The format of the PDD entry is as follows:

Characters	Content
0 - 7	PGMDIRCT.
8 - 11	Relative drum address of first PD.
12 - 15	Relative drum address of first sector beyond the PD. This is the start of the actual program area.
16 - 19	Relative drum address of the next free sector available for a PD entry.

The format of the PD entry of a source or object program is as follows:

Characters	Content
0 - 7	Name.
8 - 15	Type of program:
	OBJ␣␣␣␣␣ for object code program.
	SORS␣␣␣␣ for source code program.
16 - 19	Relative drum address of first sector of this program.
20 - 23	Relative drum address of first sector beyond this program. This is the beginning of the next program.

The format of an object code segment PD is as follows:

Characters	Content
0 - 7	Segment name.
8 - 15	SEGMENT␣
16 - 19	Relative drum address of first sector of this segment.

Each PD entry must describe a contiguous area of drum storage. The boundaries of a program must be contained within the systems area with which it is associated. One PD entry will exist for each source program in a systems area. One PD entry will exist for each object program and also each segment of an object program. The maintenance of PD entries and source and object programs will be the responsibility of AJAXF and OPUSF. The size of the area in a file to be allocated for PD entries is variable depending on the number of source programs, object programs and segments of object programs that are to be stored in a particular area.

5.0 DATA FORMAT

The first 160 characters of each sector will be reserved for a record. The last eight characters of each sector are used for control purposes.

6.0 SOURCE CODE FORMAT

6.1 SOURCE PROGRAM OR PROC HEADER SECTOR

Positions	Content
0	D(027) sector type.
1 - 3	Sector number (binary).
4 - 7	Run ID.
8 - 167	Not used (may contain comments).

6.2 SOURCE PROGRAM OR PROC SENTINEL SECTOR

Positions	Content
0	F (031) sector type.
1 - 3	Sector number (binary).
4 - 167	Not used.

The formats given in Sections 6.1 and 6.2 are produced on the drum from 80 or 90 column cards through use of AJAXF.

The 6.1 form must precede all source programs or PROCS and the 6.2 form must follow all source programs or PROCS that are being filed on the drum from cards.

6.3 SOURCE CODE SECTOR

Positions	Content
0	E (030) sector type.
1 - 3	Sector number (binary).
4 - 5	Number of source code cards in this sector (binary).
6 - 85	Characters 1 - 80 of source code.
86 - 165	Characters 1 - 80 of source code.
166 - 167	Unused.

7.0 OBJECT CODE FORMAT

7.1 PROGRAM HEADER SECTOR

Positions	Contents
0	Q (053) sector type.
1 - 3	Sector number (binary).
4 - 7	Run ID.
8 - 167	Not used (may contain comments).

This sector must precede all object programs on the drum. It is automatically generated by the PAL Drum Assembler.

7.2 R SECTOR

Positions	Content
0	R (054) sector type.
1 - 3	Sector number (binary).
4 - 5	Not used.
6 - 9	Relative address of assembly.
10 - 12	Total number of locations assigned to program.
13 - 15	Number 1 higher than highest location into which information will be loaded.
16 - 135	Not used.
136 - 139	077777777.
140 - 143	Run ID.
144 - 145	Load Key.
146 - 147	02001 (R in column 74).
148 - 167	Not used.

7.3 S SECTOR

Positions	Content
0	S (065) sector type.
1 - 3	Sector number (binary)
4 - 135	Not used.
136 - 139	077777777.
140 - 143	Segment ID.
144 - 145	Load Key.
146 - 147	01200 (S in column 74).
148 - 167	Not used.

7.4 INSTRUCTION SECTOR

Positions	Content
0	W (071) sector type.
1 - 3	Sector number (binary).
4 - 5	Not used.
6 - 25	Relocation mask.
26 - 29	Starting address.
30 - 31	Number of characters to be loaded from this sector.
32 - 143	Data to be loaded.
144 - 145	Check sum.
146 - 147	Blank.
148 - 167	Not used.

7.5 T SECTOR

Positions	Content
0	T (066) sector type.
1 - 3	Sector number (binary)
4 - 31	Not used.
32 - 36	Jump instruction to start of program.
37 - 38	Program (excluding routine header) or segment count.
39 - 143	Not used.
144 - 145	Check sum.
146 - 147	01100 (T in column 74).
148 - 167	Not used.

7.6 LDKEY SECTOR

Positions	Content
0	Any allowable key supplied by the use of the LDKEY assembler directive.
1 - 3	Sector number (binary).
4 - 5	Not used.
6 - 25	Relocation mask.
26 - 29	Starting address.
30 - 31	Number of characters to be loaded from this sector.
32 - 143	Data to be loaded.
144 - 145	Check sum.
146 - 147	Blank.
148 - 167	Not used.

7.7 PROGRAM SENTINEL SECTOR

Position	Content
0	Y (073) sector type.
1 - 3	Sector number (binary).
4 - 167	Not used.

This sector must follow all object programs on the drum. It is automatically generated by the PAL Tape Assembler.

8.0 SOFTWARE ROUTINES

8.1 INITIAL DRUM SETUP (IDMS)

The function of the Initial Drum Setup Routine is to accept parameter cards and construct a systems directory. The IDMS routine assumes that there is no data on the drum(s) to be used by the software. This routine will write the Drum Locator/Loader at the beginning of drum unit zero. IDMS must be run before the software can operate on FASTRAND drum(s). Running the IDMS routine will be the initial step in using FASTRAND drum(s).

8.2 DRUM LOCATOR LOADER (DLL)

The purpose of the Drum Locator/Loader is to locate and load a Drum Executive Routine. The Drum Locator/Loader will be loaded manually from the console by an operator. This program will accept the program identification of an object program to be loaded from a drum (normally this will be an Executive Routine), locate and load it. The Drum Locator/Loader, however, will have the ability to locate and load any absolute object program on a drum.

8.3 SYSTEMS AREA DRUM LOAD (SADL)

The purpose of this program is to load programs to a systems area on a drum from tape or cards. The systems area drum load assumes that the drum area to which it is loading contains no valid information. SADL will be used to write the systems program file on a drum. It may be used to write programs in any defined systems area.

8.4 DRUM EXECUTIVE ROUTINE (OPRF)

The Drum Executive Routine will control programs basically in the same manner the real time Executive Routine (OPR) presently does. The Drum Executive will have the ability to locate and load relative or absolute programs from any drum unit as well as from any tape unit. The user will have the ability to assemble a Drum Executive that will suit the needs of his configuration and operational needs.

8.5 PAL DRUM ASSEMBLER

The PAL Drum Assembler will accept source code input from cards, tape or drum. The intermediate phases of the assembler will require a scratch area on the drum (see Section 4.5 for scratch area definition). The object code output of an assembly will be written on the drum. The PAL Drum Assembler will operate independently of the Executive Routine.

8.6 AJAXF AND OPUSF

AJAXF AND OPUSF will be reponsible for maintaining and updating all source and/or object code routines in the systems areas. AJAXF and OPUSF will accept parameter cards directing it to delete, file, replace, print, punch or alter programs in a systems area on a drum(s). AJAXF and OPUSF will operate without an Executive Routine.

8.7 FASTRAND DUMP (FRDP)

The FASTRAND Dump Routine will dump a class of area (that which is defined in a Systems Directory entry) to tape or cards in a form that is suitable for reloading. While dumping, the user will have the ability to perform utility services on selected files within a class of area. This program will also print files and directories. FRDP will dump classes of areas that contain data files and classes of areas that define systems areas. Parameter cards will be submitted to this routine directing it to perform specific operations.

8.8 FASTRAND LOAD (FRLD)

The FASTRAND Load Routine will load classes of areas to drum(s) from cards or tape. FRLD assumes that the class that is being loaded contains no data; consequently, only that which is on a tape or cards will constitute the contents of a class that is loaded. Only classes that have been dumped using the FRDP program will be loaded by FRLD. Parameter cards control the loading of the classes to the drum(s).

8.9 DRUM SORT

The user will have the ability to generate a Drum Sort object program that is suitable for his particular needs. First Pass Own Coding will be required to deliver records to the sort. Input to the sort may be introduced from cards, tape or drum. Last Pass Own Code will accept sorted records from the sort. Output from the sort may go to cards, drum, tape or printer. The sort will require that an area on a single drum unit be reserved for intermediate processing by the sort. An object sort may run concurrently.

8.10 FILE CONTROL

The purpose of the File Control Routines is to facilitate the handling of data files on a FASTRAND Drum(s). The File Control routines will be assembled with a worker program. There are two levels of drum File Control Routines: the basic level, and the record level. Both of these levels of File Control Routines operate within the framework of the directory system.